WHAT IS CLAIMED IS:

1. A method for managing a link stack comprising the steps of:

for a first type instruction:

setting a first data value corresponding to a first address stored in said link stack in a first portion of an entry in a queue having a plurality of entries; and

for a second type instruction:

setting a current value of a first pointer into said link stack in a first register; and

setting said current value of said first pointer in a second portion of said entry in said queue.

2. The method of claim 1 further comprising the steps of:

for said second type instruction, reading from said link stack a second address stored at a stack entry at said current value of said first pointer; and

storing said second address in a second register.

3. The method of claim 1 wherein said first type instruction is a "push" type instruction an said second type instruction is a "pop" type instruction.

4. The method of claim 1 further comprising the step of, for a first type operation, setting said first data value in a third register.

5. The method of claim 4 further comprising the step of, for third type instruction, setting a second data value from said second register in a said first portion of said entry.

6. The method of claim 1 wherein said steps recited therein are performed in response to a fetch of a corresponding one of said first type instruction and said second type instruction

7. The method of claim 2 further comprising the steps of:

receiving a pointer value in response to a pipeline flush, said pointer value operable for pointing into said queue; and

setting a current pointer value of said first pointer to a value in said second portion of an entry pointed to by said pointer value.

8. The method of claim 7 further comprising the steps of:

retrieving a value from said second register; and

comparing said value from said retrieving step with a value from said second portion of said entry pointed to by said pointer value.

9. The method of claim 8 further comprising the steps of:

in response to a compare in said comparing step, comparing said current value of said first pointer with a value in said first register; and

in response to a compare of said current value of said first pointer and said value in said first register, setting said value in said second register in said link stack at a location pointed to by said current value of said first pointer decremented by one.

10. A method of managing a link stack comprising:

for a first type instruction:

setting, in a first register, a first data value from a link stack entry pointed to by a current value of a first pointer;

setting said current value of said first pointer in a second register; and

setting a second data value in said link stack at an entry pointed to by said current value of said first pointer; and

for a second type instruction:

setting said decremented value of said first pointer into a second portion of an entry in a queue having a plurality of entries.

11. The method of claim 10 further comprising the steps of:

for said first type operation:

setting a first Boolean value in a third register;

setting a current value of said second pointer in a fourth register, said second pointer operable for pointing into a queue for tracking instructions; and

for said second type operation:

decrementing a current value of said first pointer by one;

reading from said link stack a first address stored at a stack entry at said decremented current value of said first pointer; and

incrementing a current value of a second pointer, said second pointer pointing to an entry point in said queue.

12. The method of claim 11 further comprising the steps of, in response to a completion of an instruction:

comparing a current value of a third pointer and a value in said fourth register, wherein said fourth pointer is operable for deallocating an entry in said queue.

13. The method of claim 12 further comprising, in response to a compare in said comparing step, the steps of:

placing a second Boolean value in said fourth register; and

decrementing said fourth pointer.

14. The method of claim 11 further comprising the steps of:

determining if a value in said fourth register is between a current value of said second pointer and a pointer value received in response to a pipeline flush; and

determining if a current value in said third register corresponds to said first Boolean value.

15. The method of claim 14 further comprising the step of setting a current value from said first register in a link stack entry pointed to by a value from said second register in response to said determining steps evaluating logically true.

16. The method of claim 13, wherein said first Boolean value represents logical true and said second Boolean value represents logical false.

17. The method of claim 14 further comprising the step of setting a current value of said second pointer to said pointer value received in response to said pipeline flush.

18. A data processing system comprising:

a central processing unit (CPU), said CPU including:

a link stack; and

first logic operable for, for a first type instruction, setting a first data value corresponding to a first address stored in said link stack in a first portion of an entry in a queue having a plurality of entries, and for a second type instruction, setting a current value of a first pointer into said link stack in a first register, and setting said current value of said first pointer in a second portion of said entry in said queue

19. The system of claim 18 wherein said CPU further comprises, second logic operable for, for said second type instruction, reading from said link stack a second address stored at a stack entry at said current value of said first pointer, and storing said second address in a second register.

20. The system of claim 18 wherein said first type instruction is a "push" type instruction an said second type instruction is a "pop" type instruction.

21. The system of claim 18 wherein said CPU further comprises, third logic operable for, for a first type operation, setting said first data value in a third register.

22. The system of claim 21 wherein said CPU further comprises fourth logic operable for, for third type instruction, setting a second data value from said second register in a said first portion of said entry.

23. The system of claim 18 wherein said first logic sets said first data value, for said first type instruction, and sets said current value of said pointer, for said second type instruction in response to a fetch of a corresponding one of said first type instruction and said second type instruction.

24. The system of claim 19 wherein said CPU further comprises:

fifth logic operable for receiving a pointer value in response to a pipeline flush, said pointer value operable for pointing into said queue; and

sixth logic operable for setting a current pointer value of said first pointer to a value in said second portion of an entry pointed to by said pointer value.

25. The system of claim 24 wherein said CPU further comprises:

seventh logic operable for retrieving a value from said second register; and

eighth logic operable for comparing said value from said retrieving step with a value from said second portion of said entry pointed to by said pointer value.

26. The system of claim 25 wherein said CPU further comprises:

ninth logic operable for, in response to a compare in said comparing step, comparing said current value of said first pointer with a value in said first register; and

tenth logic operable for, in response to a compare of said current value of said first pointer and said value in said first register, setting said value in said second register in said link stack at a location pointed to by said current value of said first pointer decremented by one.

27. The system of claim 18 further comprising system memory coupled to said CPU, said system memory operable for storing a program of instructions including instructions of said first type and said second type.

28. A data processing system comprising:

a central processing unit (CPU), said CPU including:

a link stack;

a first register;

a second register

first logic operable for, for a first type instruction, setting, in said first register, a first data value from a link stack entry pointed to by a current value of a first pointer, setting said current value of said first pointer in said second register, and setting a second data value in said link stack at an entry pointed to by said current value of said first pointer; and

second logic operable for, for a second type instruction, decrementing a current value of said first pointer, reading from said link stack a first address stored at a stack entry at said decremented current value of said first pointer, setting said decremented value of said first pointer into a second portion of an entry in a queue having a plurality of entries, said queue operable for tracking instructions and incrementing a current value of a second pointer, said second pointer pointing to an entry point in said queue.

29. The system of claim 28 wherein said CPU further comprises:

a third register;

a fourth register;

third logic operable for, for said first type operation, setting a first Boolean value in said third register, and setting a current value of a second pointer in said fourth register, said second pointer operable for pointing into said queue; and wherein said second logic is further operable for, for said second type instruction.

30. The system of claim 29 wherein said CPU further comprises:

fourth logic operable for, in response to a completion of an instruction, comparing a current value of a third pointer and a value in said fourth register, wherein said fourth pointer is operable for deallocating an entry in said queue.

31. The system of claim 30 wherein said CPU further comprises, fifth logic operable for, in response to a compare in said fourth logic, placing a second Boolean value in said fourth register, and decrementing said fourth pointer.

32. The system of claim 29 wherein said CPU further comprises:

sixth logic operable for determining if a value in said fourth register is between a current value of said second pointer and a pointer value received in response to a pipeline flush; and

seventh logic operable for determining if a current value in said third register corresponds to said first Boolean value.

33. The system of claim 32 wherein said CPU further comprises eighth logic operable for setting a current value from said first register in a link stack entry pointed to by a value from said second register in response to said sixth and seventh logic outputting a logical true value.

34. The system of claim 31 wherein said first Boolean value represents a logical true value and said second Boolean value represents a logical false value.

35. The system of claim 32 wherein said CPU further comprises ninth logic

operable for setting a current value of said second pointer to said pointer value received in response to said pipeline flush.

36. A method for managing a link stack comprising the steps of:

for a first type instruction:

setting a first address in a first portion of an entry in a queue having a plurality of entries; and

setting a current value of a first pointer into said link stack in a second portion of said entry in said queue; and.

for a second type instruction:

setting a second current value of said first pointer in said second portion of said entry in said queue.

37. The method of claim 36 further comprising the steps of:

for said second type instruction, reading from said link stack a second address stored at a stack entry at said second current value of said first pointer.

38. The method of claim 36 wherein said first type instruction is a "push" type instruction an said second type instruction is a "pop" type instruction.

39. The method of claim 36 wherein said steps recited therein are performed in response to a fetch of a corresponding one of said first type instruction and said second type instruction.

40. The method of claim 36, for a third type instruction comprising the steps of:

receiving a pointer value from, said pointer value operable for pointing into said queue; and

setting a current pointer value of said first pointer to a value in said second portion of an entry pointed to by said pointer value.

41. The method of claim 40 wherein said third type instruction is a "flush" type instruction.

42. A data processing system comprising:

a central processing unit (CPU), said CPU including:

a link stack;

a queue having a plurality of entries, said queue operable for tracking instructions; and

first logic operable for, for a first type instruction, setting a first address into a stack entry in said link stack at a current value of a first pointer and setting a current value of said first pointer into a first portion of an entry in said queue.

43. The system of claim 42, wherein said CPU further comprises, second logic operable for, for a second type instruction, setting a second current value of said first pointer in said second portion of said entry in said queue.

44. The system of claim 43, wherein said second logic, for said second type instruction, is further operable to read from said link stack a second address stored at a stack entry at said second current value of said first pointer.

45. The system of claim 42, wherein said first type instruction is a "push" type instruction.

46. The system of claim 43, wherein said second type instruction is a "pop" type instruction.

47. The system of claim 42, wherein said CPU further comprises, third logic operable for, for a third type operation, receiving a pointer value, said pointer value operable for pointing into said queue and setting a current pointer value of said first pointer to a value in said second portion of an entry pointed to by said pointer value.

48. The system of claim 47, wherein said third type instruction is a "flush" type instruction.